




TemplateExample		 Edit ? - 	
CALLABLE TABLE		PORTLET ACTION DEMO	
THIS IS AN EXAMPLE PORTLET:			
OPERATING IN VIEW MODE.			
<div>PORTLET ACTION DEMO</div>			
LABEL EXAMPLE:		<div></div>	
<div> SUBMIT</div>			
<div>NOTES</div>			
DEVELOPERS MUST FOLLOW THE STANDARDS: → <u>USABILITY REQUIREMENTS CHECKLIST</u>			

FIG. 1
RELATED ART

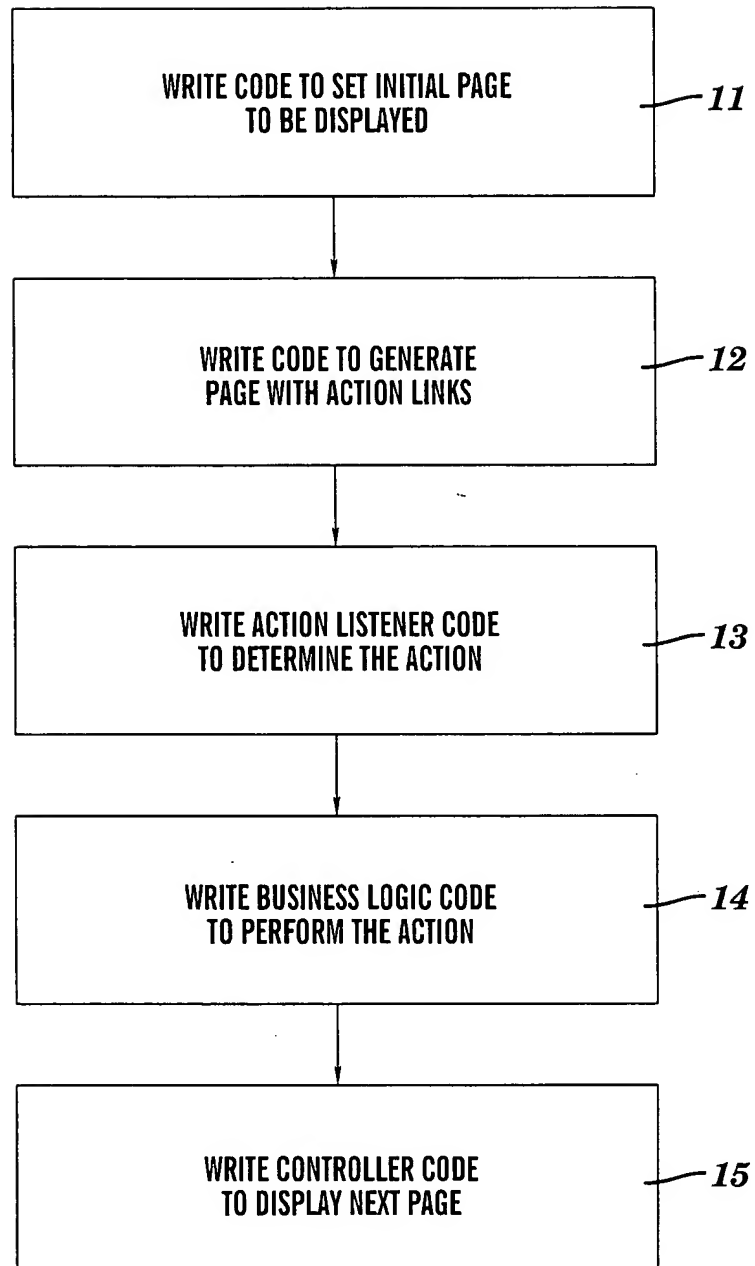


FIG. 2

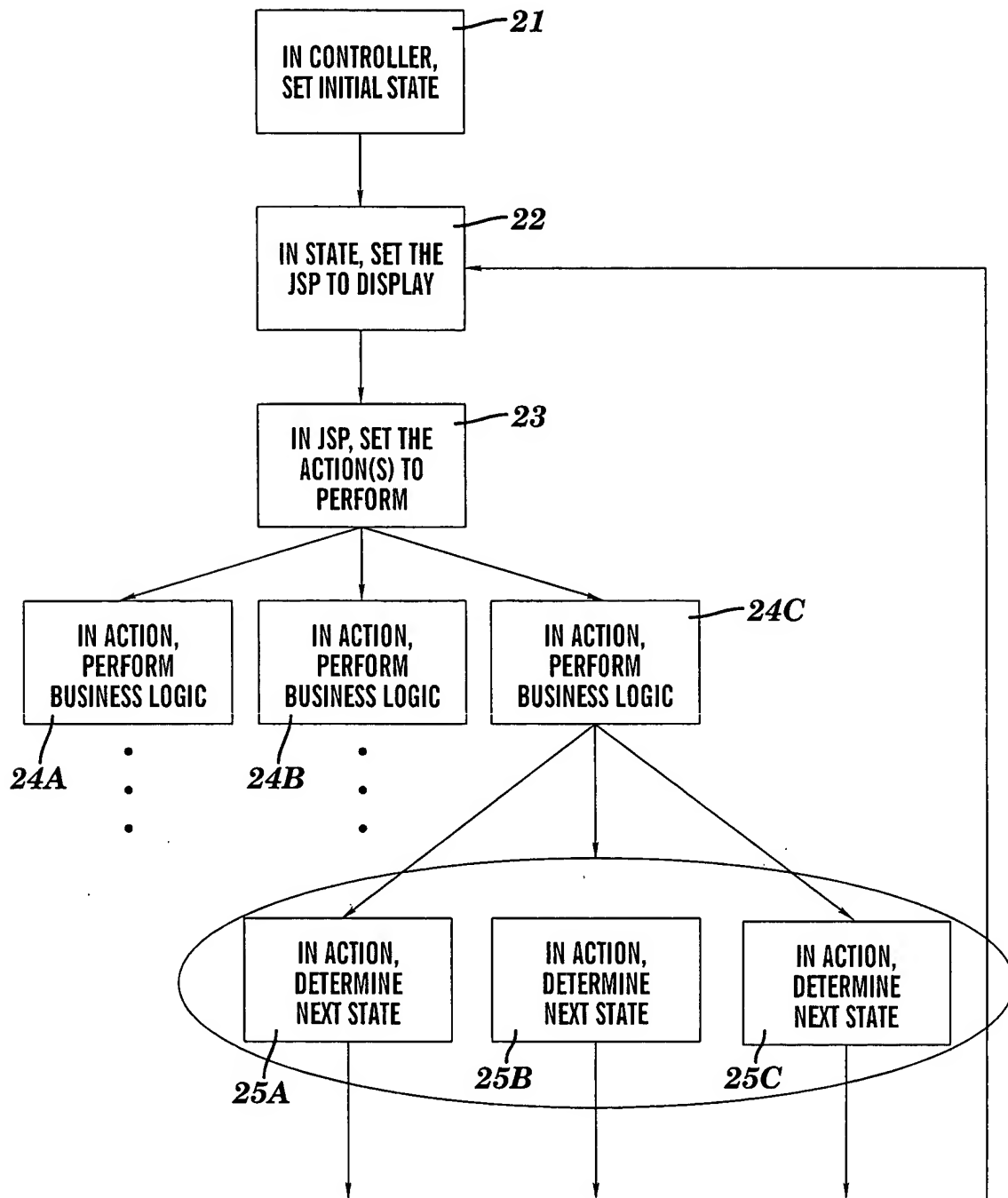


FIG. 3

PORTLET TEMPLATE BY RENAMING AND CUSTOMIZING THE TemplatePortlet AND THE TemplateControllerForHtml, THE DEVELOPER CAN FOCUS, MAINLY, ON THE STATE AND ACTION CLASSES.

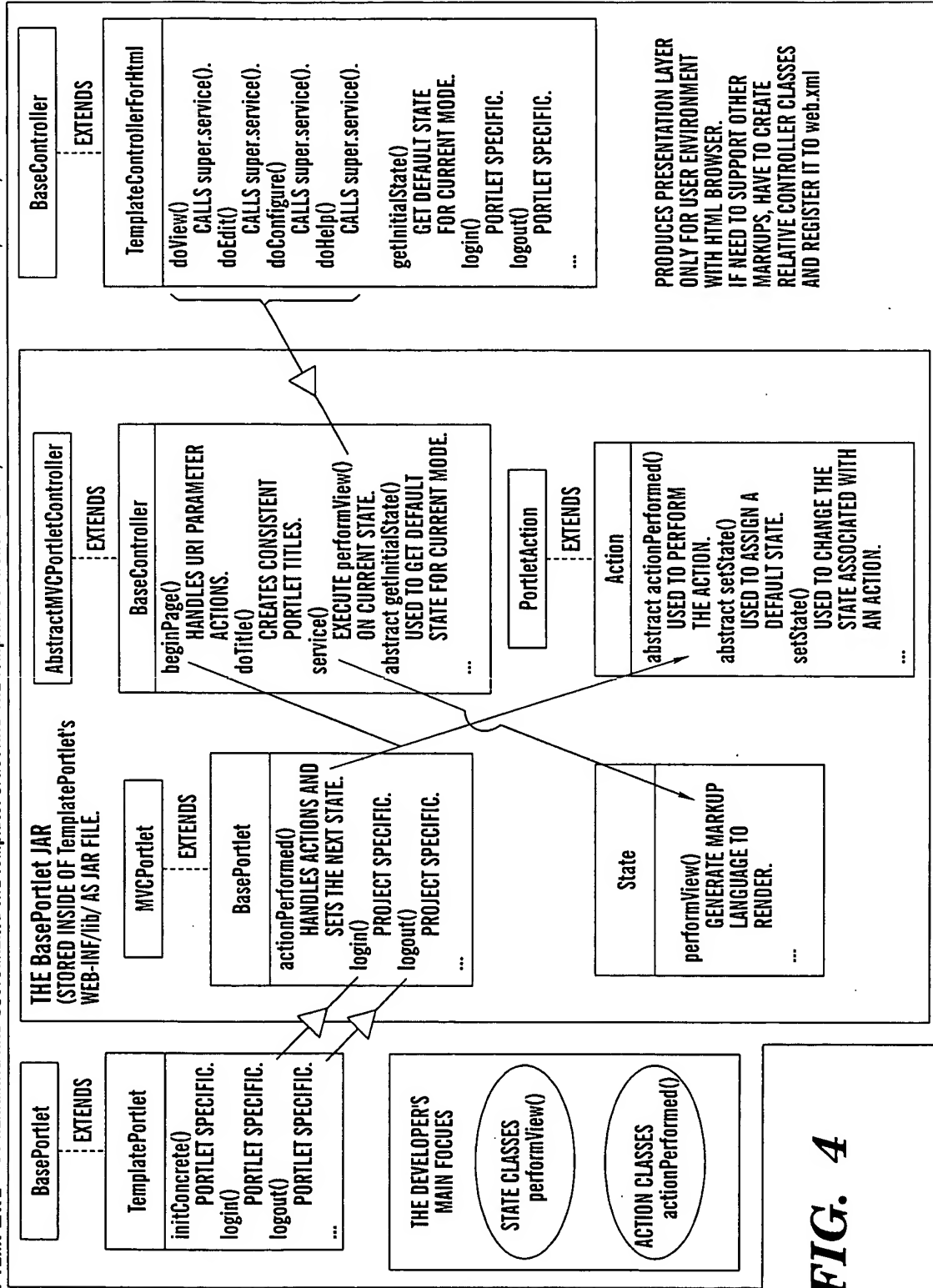


FIG. 4

Portlet Base Source (Part A)

```
public class BasePortlet extends MVCPortlet {
.
.
.
/*****
*   Public Method Name: actionPerformed
*
*   Purpose:
*       This method is the action listener. It is
*       responsible for processing the information
*       that the user has entered in the portlet.
*
*   @param ActionEvent
*       Event which represents the specific user
*       action
*
*   @return none
*
*   @throws PortletException
*****/
public void actionPerformed(ActionEvent event) throws
PortletException {

    // Pull the portlet request out of the event.
    PortletRequest request = event.getRequest();

    try {
        // If we don't have a request, I think
        // something is really wrong. We
        // log an error.
        if (request == null) {
            throw new Exception(
                "In "
                + this.getClass().getName()
                + ".actionPerformed(), request is
                null.");
        }

        // Execute the perform action method for the event
        Action action = (Action) event.getAction();

        action.actionPerformed(
            baseClassService,
            request,
            getPortletConfig());
    }
}
```

FIG. 5A

Portlet Base Source (Part B)

```
        // Set the default state for this action.
        action.setState(request);
    } catch (Exception e) {

        // Rethrow PortletExceptions
        if (e instanceof PortletException) {
            throw (PortletException) e;
        }

        // Defer all other Exception handling
        postActionException(request, e);
    }
    super.actionPerformed(event);
}
.
.
.
} //end Class
```

FIG. 5B

Template Controller and Base Controller Source (Part A)

```
public class BaseController extends MVCPortlet {
.
/*****
*   Public Method Name: service
*   Purpose:
*       This method is the method which is called when
*       the user selects to view the portlet. It is
*       responsible for rendering appropriately.
*
*   @param PortletRequest
*       Portlet request.
*   @param PortletResponse
*       Portlet response from view.
*   @return none
*   @throws PortletException
*   @throws IOException
*****/
public void service(PortletRequest request,
PortletResponse response)
    throws PortletException, IOException {

    try {
        // Handle deferred exceptions
        processActionException(request);

        // Get the portlet state object from session, if
        // not there get the initial state for the mode.
        PortletSession session = request.getPortletSession();
        State nextState =
            (State) session.getAttribute(
                request.getMode().toString() +
                State.STATE_OF_WORKFLOW);
        if (nextState == null) {
            nextState = getInitialState(request.getMode());
            request.getPortletSession().setAttribute(
                request.getMode().toString() +
                State.STATE_OF_WORKFLOW, nextState);
        }
        // Dispatch to state
        nextState.performView(
            baseClassService,
            request,
            response,
            getPortletConfig());
    } catch (Exception e) {

        // Display any generated exceptions
        displayException(request, response, e);
    }
}
```

FIG. 6A

Template Controller and Base Controller Source (Part B)

```
    }  
  }  
  } //end Class  
  
public class TemplateControllerForHtml extends  
MSCBaseController  
{  
  /******  
  * Purpose:  
  * Below API method are handled as states.  
  *  
  * @param PortletRequest  
  * @param PortletResponse  
  *  
  * @return none  
  *  
  * @throws PortletException  
  * @throws IOException  
  *****/  
  public void doConfigure(PortletRequest request,  
    PortletResponse response)  
    throws PortletException, IOException {  
    service(request, response);  
  }  
  
  public void doEdit(PortletRequest request, PortletResponse  
    response)  
    throws PortletException, IOException {  
    service(request, response);  
  }  
  
  public void doHelp(PortletRequest request, PortletResponse  
    response)  
    throws PortletException, IOException {  
    service(request, response);  
  }  
  
  public void doView(PortletRequest request, PortletResponse  
    response)  
    throws PortletException, IOException {  
    service(request, response);  
  }  
}  
//end Class
```

FIG. 6B

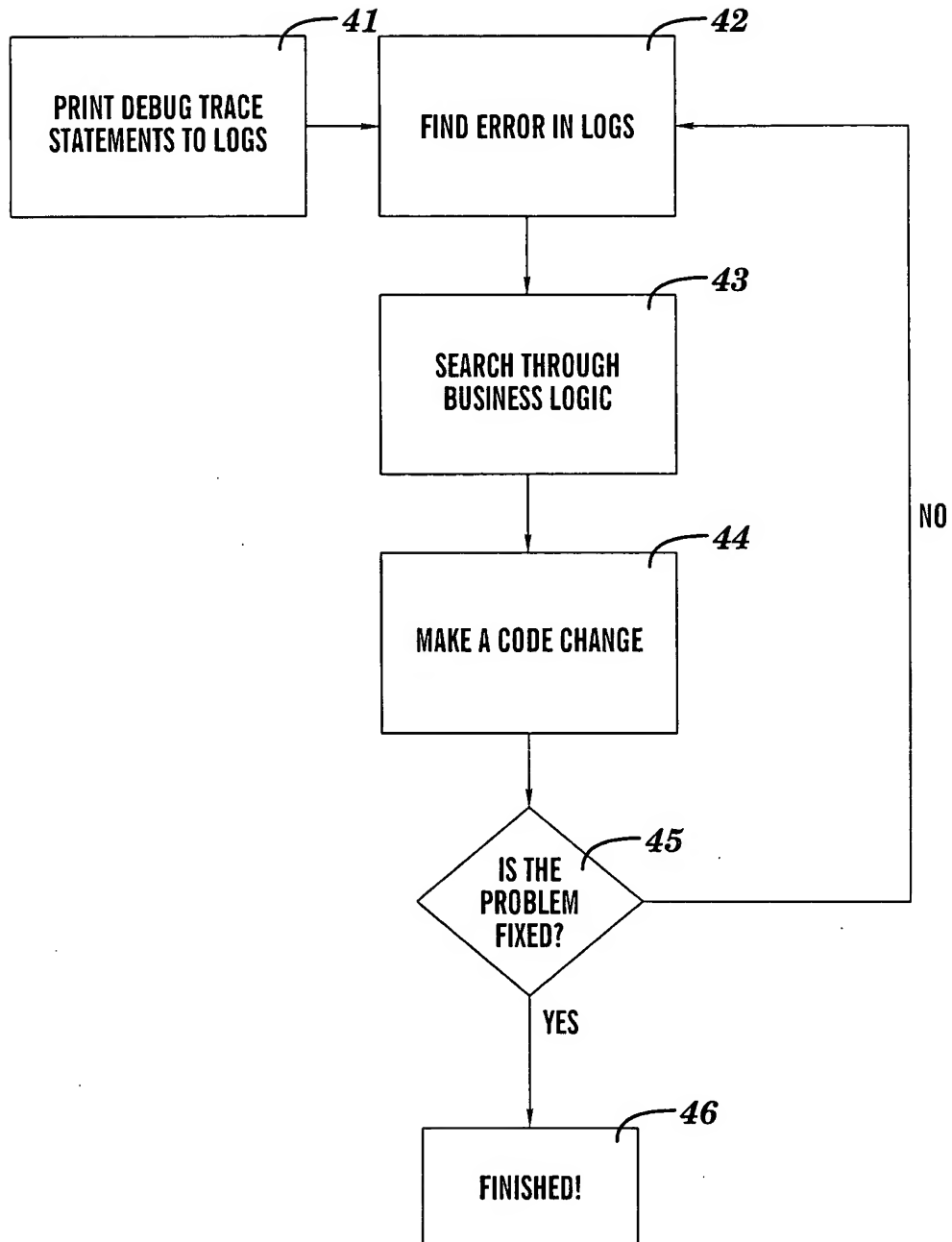


FIG. 7

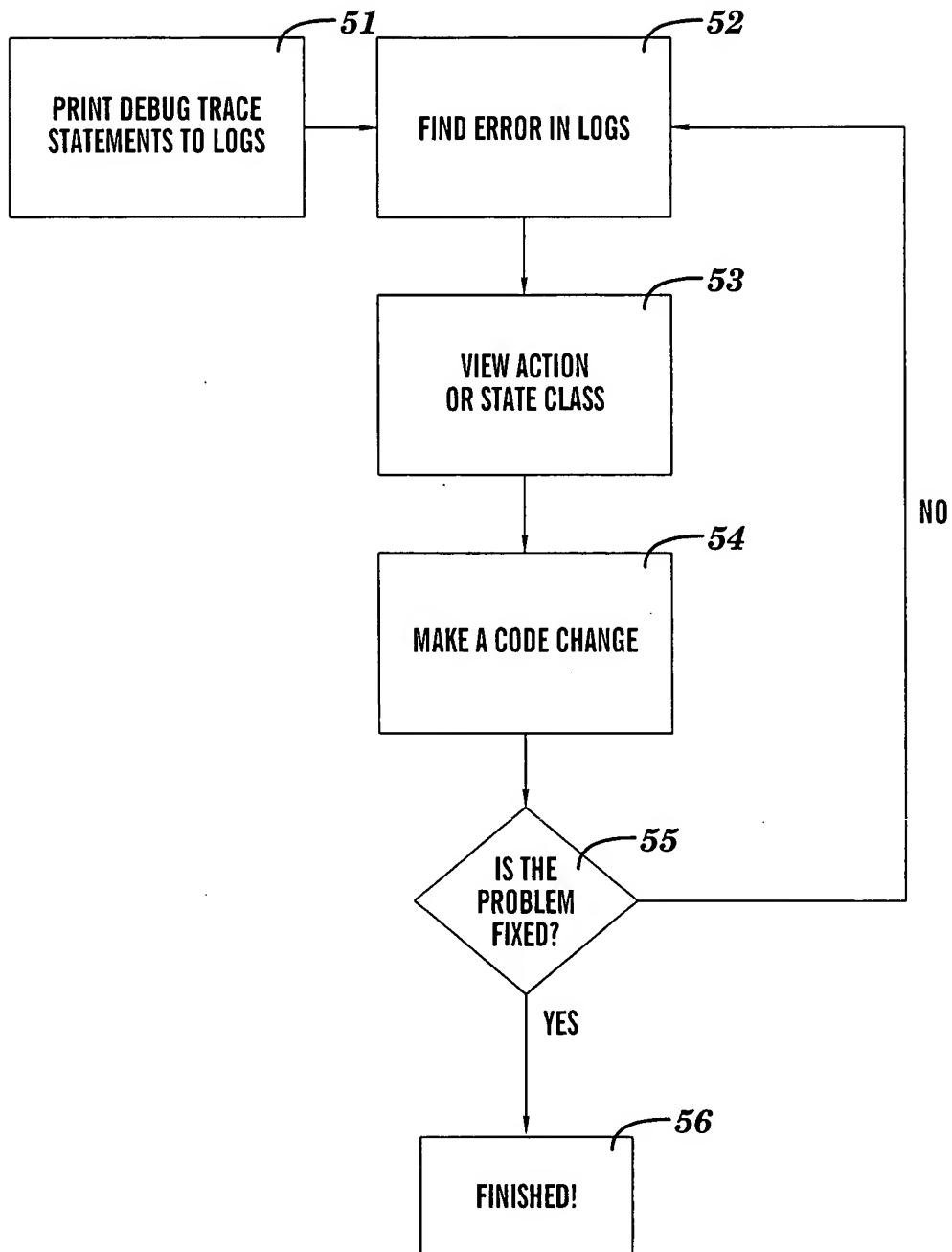


FIG. 8

**CHANGE/REDESIGN PORTLET FLOW
WITHOUT THE PORTLET TEMPLATE.**

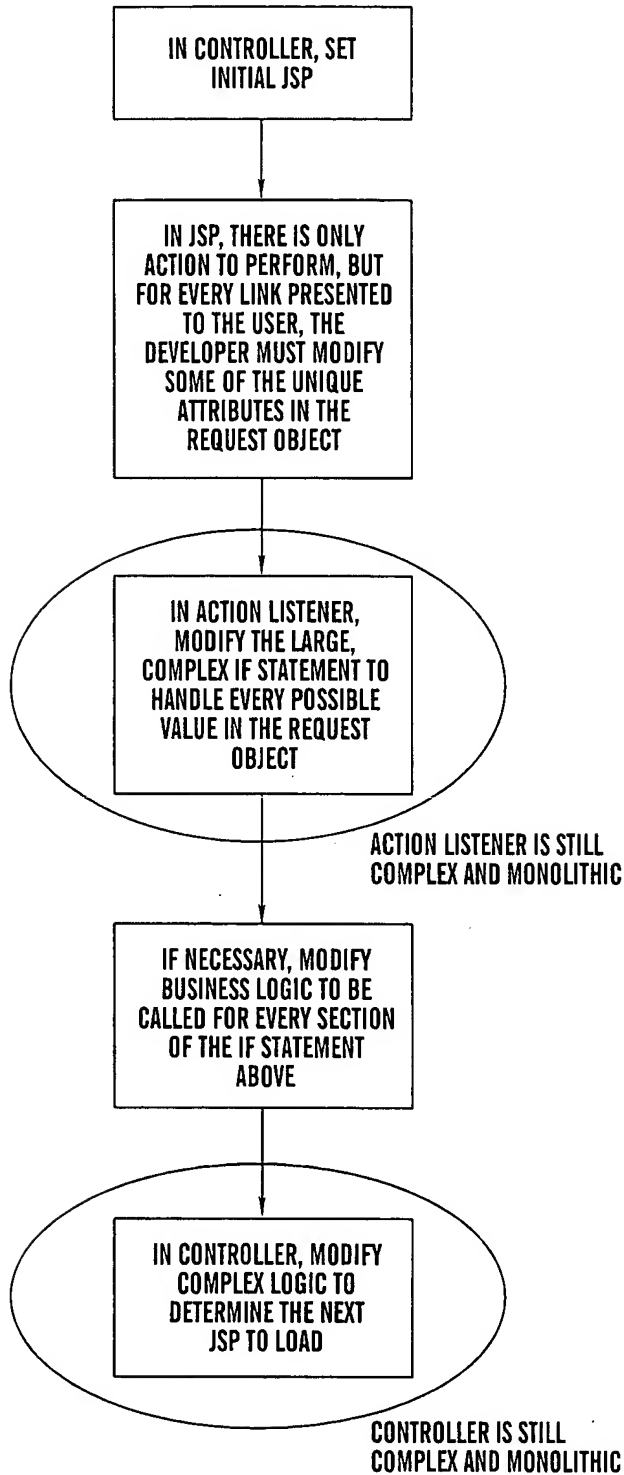


FIG. 9A

**CHANGE/REDESIGN PORTLET FLOW
WITH THE PORTLET TEMPLATE.**

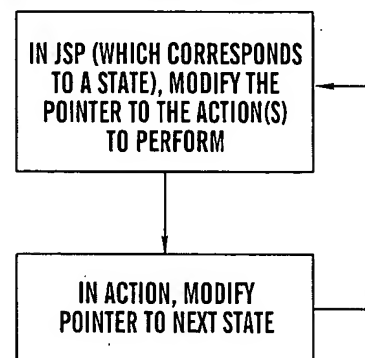


FIG. 9B

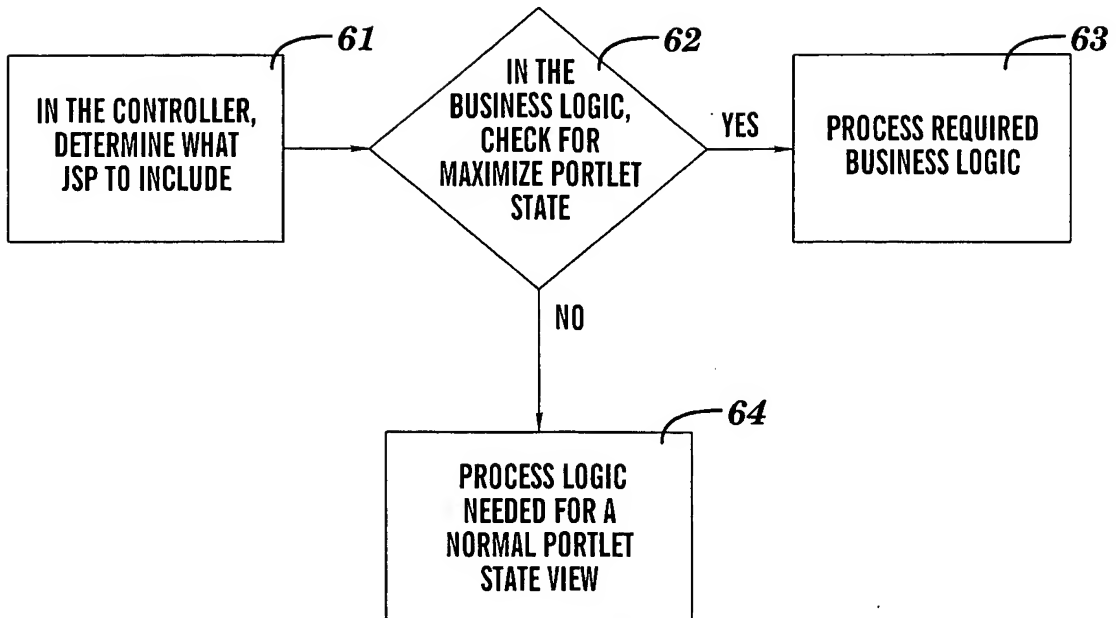


FIG. 10

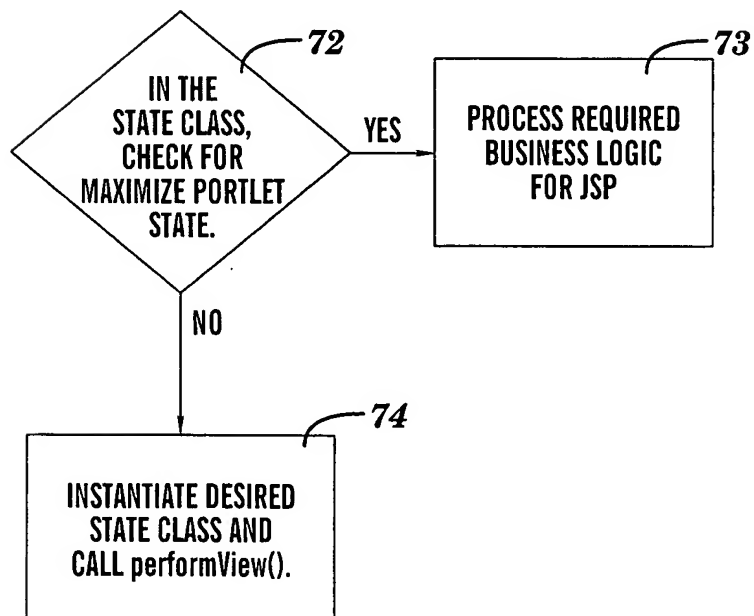


FIG. 11

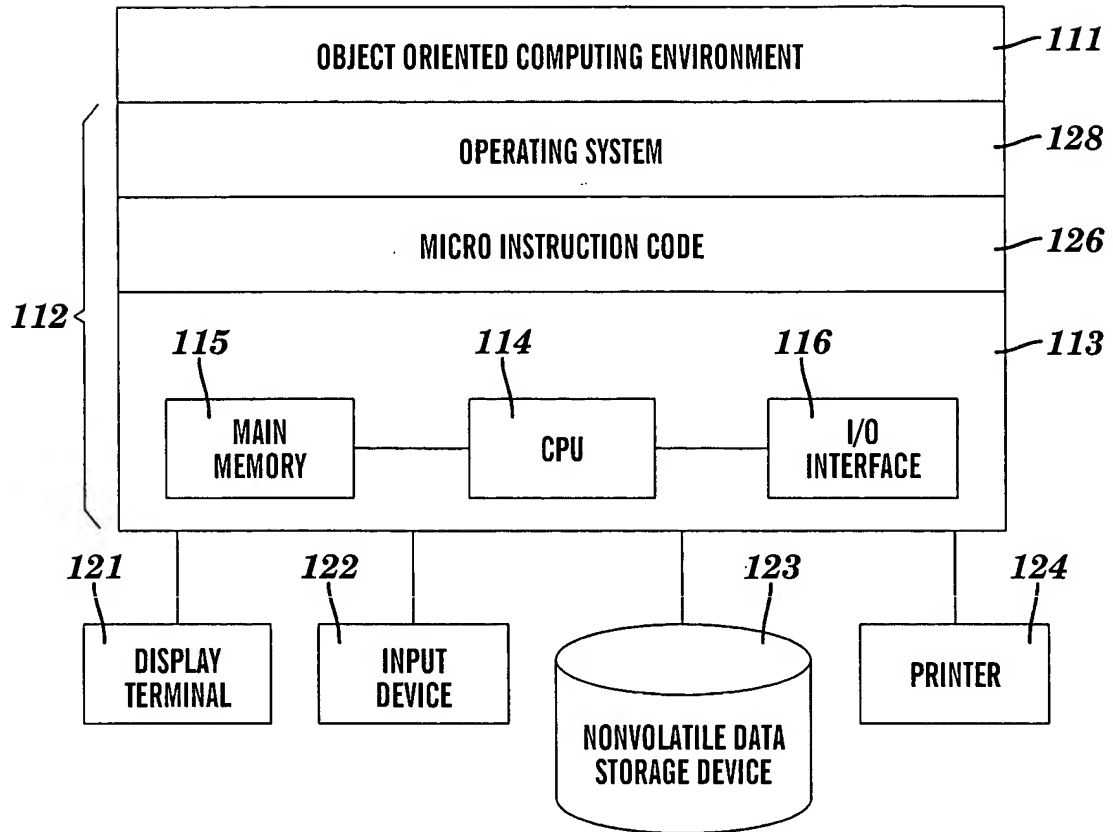


FIG. 12

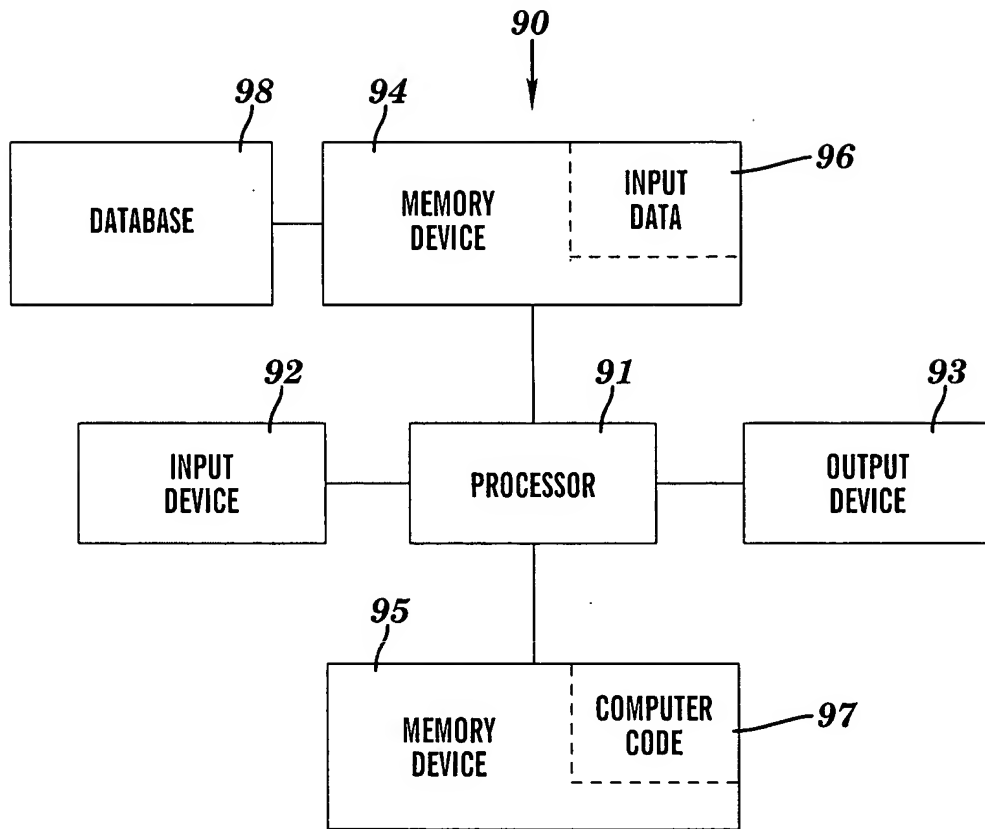


FIG. 13